## **Journal Program: Submission Comment**

**Project Outline**

**Program Class:**

* Description: Acts as the main entry point for the program.
* Members:
* Main(string[] args): Entry point method that handles the menu and program flow.

**Journal Class:**

* Description: Manages the journal as a collection of JournalEntry objects.
* Members:
* List<JournalEntry> entries: Stores the list of journal entries.
* List<string> prompts: Stores the list of prompts for journal entries.
* AddEntry(JournalEntry entry): Adds a new entry to the journal.
* DisplayEntries(): Displays all journal entries to the console.
* GetRandomPrompt(): Generates a random prompt from the list of prompts.
* SaveToFile(string fileName): Saves the journal entries to a file.
* LoadFromFile(string fileName): Loads journal entries from a file.

**JournalEntry Class:**

* Description: Represents a single journal entry.
* Members:
* string Date: Stores the date of the entry.
* string Prompt: Stores the prompt for the entry.
* string Response: Stores the response for the entry.
* JournalEntry(string date, string prompt, string response):

Constructor to initialize a new journal entry.

**Program Flow:**

The program follows a menu-driven interface that allows the user to perform various actions:

**1 Write a New Entry:**

* The program generates a random prompt from the list of existing prompts.
* The user enters their response to the prompt.
* A new JournalEntry object is created with the current date, generated prompt, and user response.
* The entry is added to the journal.

**2 Display Journal Entries:**

* The user provides the file name of the saved journal.
* The journal entries are loaded from the specified file.
* All journal entries are displayed on the console.

**3 Save Journal to a File:**

* The user specifies the file name to save the journal.
* The journal entries are saved to the specified file.

**4 Load Journal from a File:**

* The user provides the file name to load the journal.
* The journal entries are loaded from the specified file.

**5 Exit:**

* The program terminates.

**Exceeding the Requirements:**

In addition to the core requirements, the following features were implemented to enhance the program:

* Random prompts: The program generates a random prompt for each new entry, providing users with varied prompts for their journaling.
* File management: The program allows users to save and load journal entries from different files, providing flexibility in managing multiple journal files.
* Error handling: The program includes error handling to handle invalid user inputs and provide appropriate feedback and instructions.
* Separation of concerns: The program utilizes separate classes for different functionalities, promoting code organization, reusability, and maintainability.
* Feel free to customize and expand upon this draft submission details to fit the specific requirements and achievements of your journal program.